Iris Flowers Classification ML Project

letsgrowmore Internship

task-1

DINESH CHANDRA GADDAM

Gmail : dineshsunnygaddam2002@gmail.com

In [57]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
from sklearn import preprocessing
%matplotlib inline
```

In [58]:
```
df=pd.read_excel(r"S:\documents\all courses\data_sets\iris.xlsx")
```

In [59]:
```
df.head()
```

Out[59]:

	length_sepal	width_sepal	length_petal	width_petal	Class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [60]:
```
df.columns
```

Out[60]:
```
Index(['length_sepal', 'width_sepal', 'length_petal', 'width_petal', 'Class'], dtype='object')
```

In [61]:
```
df.shape
```

Out[61]:
```
(150, 5)
```

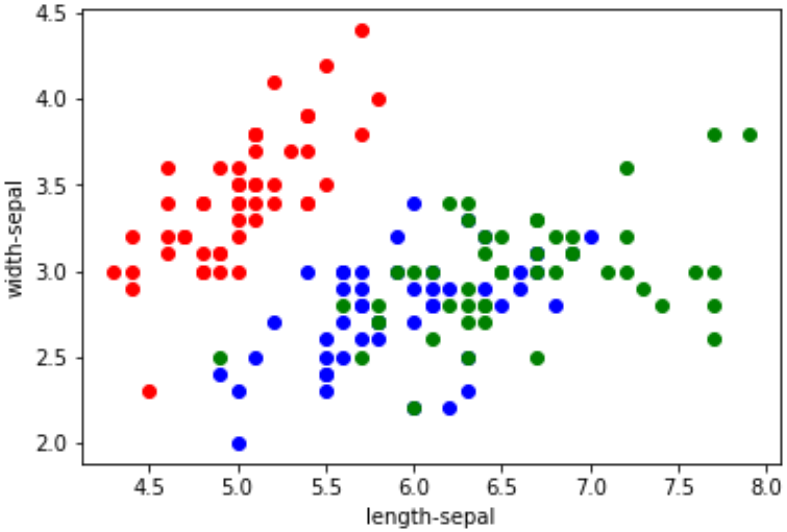
In [62]:
```
df.describe()
```

Out[62]:

	length_sepal	width_sepal	length_petal	width_petal
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

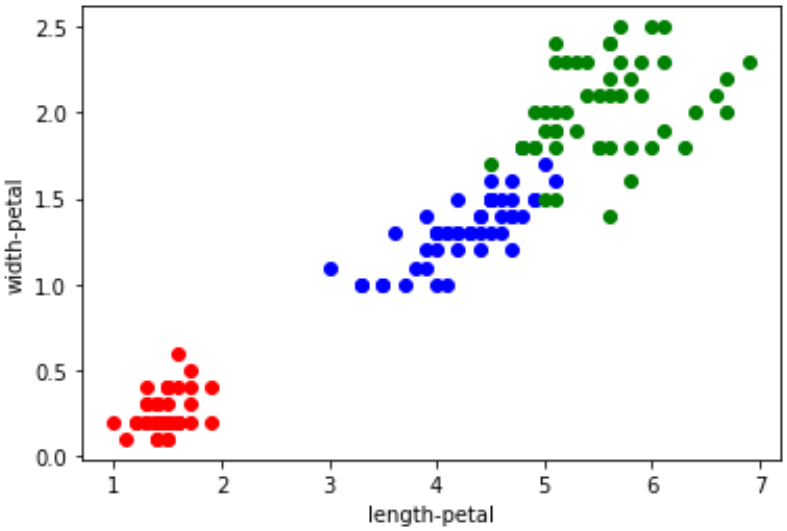
plotting via class

In [63]:
```
colors=['red','blue','green']
Class=['Iris-setosa','Iris-versicolor','Iris-virginica']
for i in range(3):
    dff=df[df['Class']==Class[i]]
    plt.scatter(x=dff['length_sepal'],y=dff['width_sepal'],color=colors[i])
    plt.xlabel('length-sepal')
    plt.ylabel('width-sepal')
```



In [64]:

```
colors=['red','blue','green']
Class=['Iris-setosa','Iris-versicolor','Iris-virginica']
for i in range(3):
    dff=df[df['Class']==Class[i]]
    plt.scatter(x=dff['length_petal'],y=dff['width_petal'],color=colors[i])
    plt.xlabel('length-petal')
    plt.ylabel('width-petal')
```



In [65]:
```
df['Class'].value_counts()
```

Out[65]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Class, dtype: int64
```

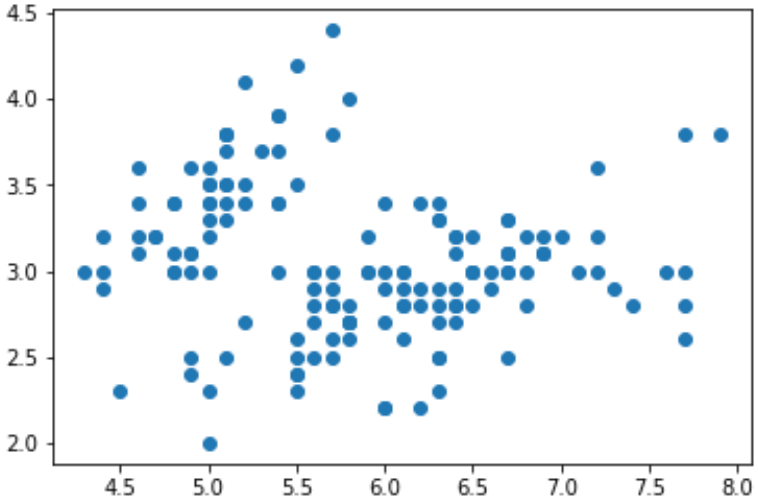
Describing X,Y for train and split data KNN

In [66]:
```
X=df[['length_sepal', 'width_sepal', 'length_petal', 'width_petal']]
Y=df['Class']
```

plotting length of sepal vs width of sepal

In [67]:
```
plt.scatter(x=df['length_sepal'],y=df['width_sepal'])
```

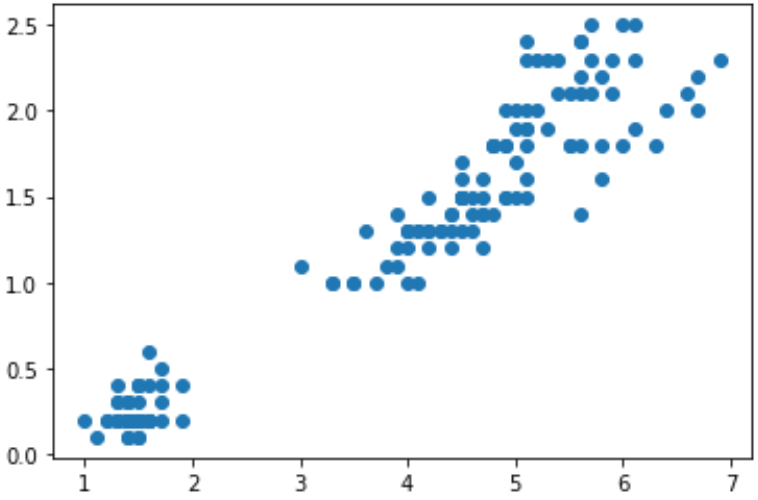
Out[67]:
```
<matplotlib.collections.PathCollection at 0x17cd8a29e20>
```



plotting length of petal vs width of petal

In [68]:
```
plt.scatter(x=df['length_petal'],y=df['width_petal'])
```

Out[68]:
```
<matplotlib.collections.PathCollection at 0x17cd8a92fd0>
```



let's train and test data using

In [69]:

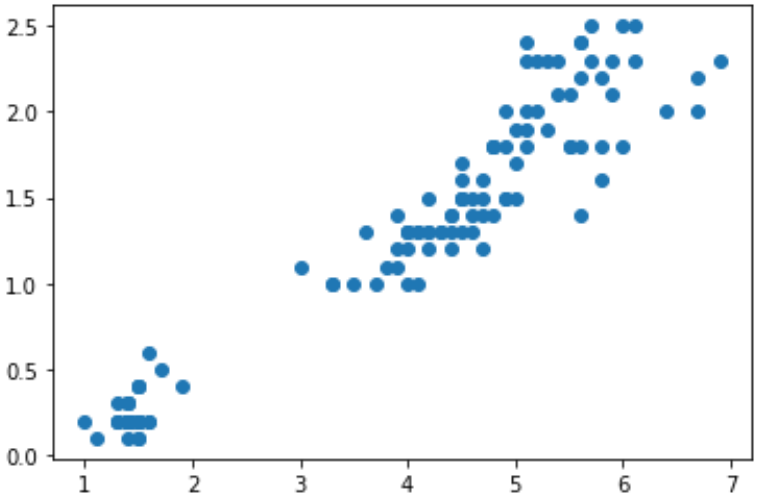
```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
import seaborn as sns
```

In [70]:
```
X_train, X_test, y_train, y_test = train_test_split( X, Y, test_size=0.2, random_state=4)
print ('Train set:', X_train.shape,  y_train.shape)
print ('Test set:', X_test.shape,  y_test.shape)
```

```
Train set: (120, 4) (120,)
Test set: (30, 4) (30,)
```

In [71]:
```
plt.scatter(x=X_train['length_petal'],y=X_train['width_petal'])
```

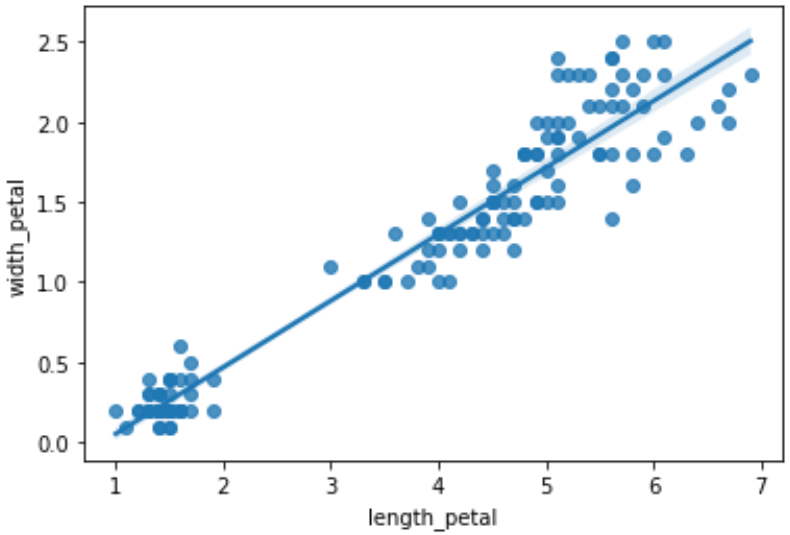
Out[71]:
```
<matplotlib.collections.PathCollection at 0x17cd8af8d60>
```



regplot for len,wid of petal

In [72]:
```
sns.regplot(x='length_petal', y="width_petal", data=df)
```

Out[72]:
```
<AxesSubplot:xlabel='length_petal', ylabel='width_petal'>
```



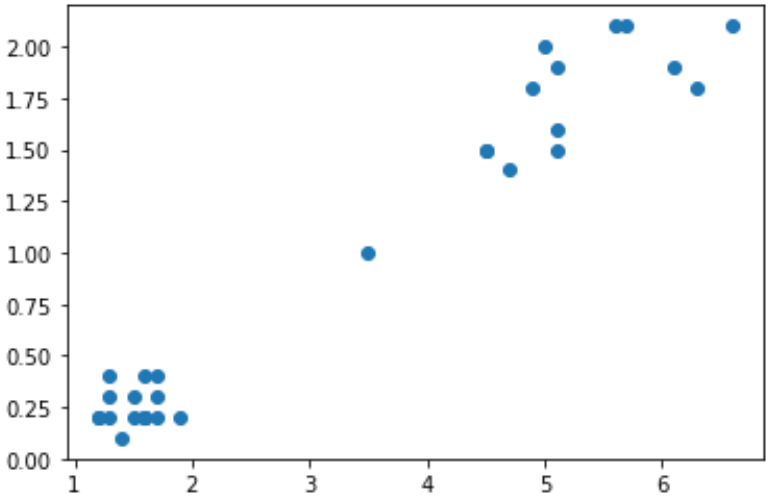
In [73]:
```
df[['length_petal',"width_petal"]].corr()
```

Out[73]:

	length_petal	width_petal
length_petal	1.000000	0.962865
width_petal	0.962865	1.000000

In [74]:
```
plt.scatter(x=X_test['length_petal'],y=X_test['width_petal'])
```

Out[74]:
```
<matplotlib.collections.PathCollection at 0x17cd8bc8a00>
```



In [75]:

```
k = 4
#Train Model and Predict
neigh = KNeighborsClassifier(n_neighbors = k).fit(X_train,y_train)
yhat = neigh.predict(X_test)
print("Train set Accuracy: ", metrics.accuracy_score(y_train, neigh.predict(X_train)))
print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat))
```

```
Train set Accuracy:  0.975
Test set Accuracy:  0.9666666666666667
```

I think this KNN accuracy is not bad

Thank You

In [ ]: